Community Detection in Social Networks

Soumyakant Priyadarshan

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India

Community detection in social networks

Thesis submitted in

June 2013

to the department of

Computer Science and Engineering

of

National Institute of Technology Rourkela

in partial fulfillment of the requirements

for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Soumyakant Priyadarshan

[Roll: 109CS0176]

with the supervision of

Prof. K.Sathyababu

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Odisha, India.

June 11, 2013

Certificate

This is to certify that the thesis entitled, ***COMMUNITY DETECTION IN SOCIAL NETWORKS*** submitted by ***SOUMYAKANT PRIYADARSHAN(109CS0176)*** in partial fulfillment of the requirements for the completion of Bachelor of Technology Degree in Computer Science and Engineering at the National Institute of Technology, Rourkela is an authentic work carried out by them under my supervision and guidance .To the best of my knowledge, Neither this thesis or any part or it has been submitted for any degree or diploma award elsewhere.

Prof.K.Sathyababu
Assistant Professor
Department of Computer Science and Engineering
NIT Rourkela

Acknowledgment

With great satisfaction and pride I present my thesis on the project under the Research Project paper during Final Year, for partial fulfillment of my Bachelor of Technology degree in Computer Science and Engineering at NIT Rourkela.

I am thankful to Prof. K.Sathyababu for being the best guide and advisor for this research work in every field I have taken to complete my requirement. His ideas and inspirations have helped me make this nascent idea of mine into a fully-fledged project. Without presence of him I may never had tasted the flavor in a research work.

Again I am thankful to my batch-mates to support me in my implementation part sometime. I am also grateful to all the professors of my department for being a constant source of inspiration and motivation during the course of the project.

I would like to dedicate this project to my parents, who always stood by me in each and every point of my life.

So I am thankful again to all who are being a part of my Final year research project.

Soumyakant Priyadarshan

Abstract

A social network is a collection of nodes representing individuals or organisations with dyadic or binary relationship between them. It is usually represented by a graph G(V,E) where V is the set of vertices representing the individuals participating in the network and E is the set of edges representing the interactions between the vertices. Examples of social network can be given as scientists co-authoring a paper, employees of a company working on a common project, etc. A community represents a group of individuals such that the frequency of interactions within the group is more than that of the interactions between the groups. Community detection problem refers to the problem of finding such groups in real world social networks. A number of methods to address this problem have been proposed earlier, and Newman distinguishes these into two categories: bottom-up sociological approaches and top-down computer science approaches. Modularity is a property of the network that measures when the division is good, in the sense that there are many edges within the community and only a few between them. In modularity based algorithms, each node of the graph is considered as an individual community and the communities are joined iteratively based on the increase in modularity caused by their joining. The ones producing maximum change in modularity are joined. There are few drawbacks associated with modularity based methods such as they require information regarding the entire structure of the network which is not possible to determine in case of vast real world networks. Also modularity optimization methods are not able to determine the overlapping communities. In order to detect overlapping communities clique percolation can be used . Clique percolation is based on the assumption that a community consists of fully connected subgraphs and detects overlapping communities by searching for adjacent cliques. But it is a hard method to implement well due to difficulty of producing intermediate representations of percolating structures. In this project we implement the k-clique percolation method using a Clique matrix and binary matrix inorder to store the intermediate percolating structure with an aim of simplifying the implementation.

Contents

List of Figures

–

Chapter 1

Introduction

1.1 Introduction to social network analysis

1.1.1 Social Network

The basic idea of a social network is very simple. A social network can be considered as a set of nodes representing individuals or oraganisations with a dyadic or binary relations between them. It is usually represented by a graph G(V,E) where V is the set of nodes representing the individuals and E is the set of edges representing the interactions between them. Few examples of social network are Scientists co-authoring a paper, employees of a company working on a common project, etc.

1.1.2 Social Network Analysis

SNA refers to the process of extracting information from a social network regarding the individuals participating in it through mapping and measuring of relationship and flows between the nodes that may represent people, groups, organizations, computers, URLs and other connected information/knowledge entities.

1.1.3 Purpose of SNA

- To make sense out of social network that is to extract information about the individuals from the network they participate in.
- To find the structure of social networks.
- Usefull in understanding the evolution of social networks.
- To discover complex communication patterns, characteristic features.

1.1.4 Importance of Social Network Analysis

- Information sharing.
- Marketing in e-commerce and e-business.
- Determine influential entities.

- Build effective social and political campaign.
- Predict future events.
- Tracking terrorists.
- Location based crowd sourcing.

1.2 Introduction to community detection

1.2.1 Community

It is formed by individuals such that those within a group interact with each other more frequently than with those outside the group. A network community (also sometimes referred to as a module or cluster) is typically thought of as a group of nodes with more and/or better interactions amongst its members than between its members and the remainder of the network.

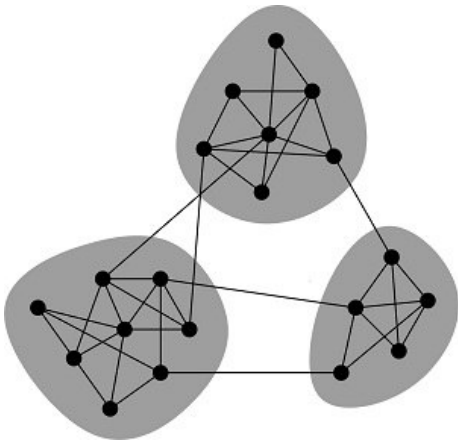

FIGURE 1.1: Community Structure

1.2.2 Community Detection

It is the process of discovering groups in a network where individuals group memberships are not explicitly given. The problem of cluster or community detection in real world graphs that involves large social networks, webgraphs and biological networks is a problem of considerable practical interest and has recieved a lot of

attention recently. To extract such sets of nodes one typically chooses an objective function that captures the above intuition of a community as a set of nodes with better internal connectivity than external connectivity. Then, since the objective is typically NP-hard to optimize exactly, one employs heuristics or approximation algorithms to find sets of nodes that approximately optimize the objective function and that can be understood or interpreted as real communities. Alternatively, one might define communities operationally to be the output of a community detection procedure, hoping they bear some relationship to the intuition as to what it means for a set of nodes to be a good community. Once extracted, such clusters of nodes are often interpreted as organizational units in social networks, functional units in biochemical networks, ecological niches in food web networks, or scientific disciplines in citation and collaboration networks

1.2.3 Purpose of community detection

- Understanding the interactions between people.
- Visualizing and navigating huge networks.
- Forming the basis for other tasks such as data mining.
- Social networks often include community groups based on common location, interests, occupation, etc. Communities are present in metaboillic networks based on functional groupings. Communities are formed in citation networks based on research topic. By identifying these sub-structures within a network can provide knowledge about how network function and topology affect each other.

1.3 Communities in social media

1.3.1 Two types of groups in social media

- Explicit Groups: formed by user subscriptions
- Implicit Groups: implicitly formed by social interactions

1.3.2 Is it necessary to extract groups based on network topology?

- All Social media websites do not provide community platform
- All people do not want to make effort to join groups. Through community extraction communitites can be suggested to people based on their interests.
- Groups in the real world change dynamically.
- Besides social media websites it is essential to extract communities in other networks such as citation networks, World Wide Web, metaboillism networks for various practical purposes.

1.3.3 Importance of network interaction

- Rich information about the relationship between users can be obtained through analysing network interaction which can complement other kinds of information, e.g. user profile
- It Provides basic information that are essential for other tasks, e.g. recommendation
- Analysing Network interaction helps in network visualization and navigation.

Chapter 2

Literature Review

2.1 Spectral bisection

A social network is usually represented by an undirected graph. The Laplacian of an undirected graph G with n vertices is given by n n symmetric matrix L. The diagonal element L_{ii} of the matrix L represents the degree of vertex i, and off-diagonal element L_{ij} is 1 if vertices i and j are connected in the given graph and zero otherwise .So it can be deduced that L = D A, where D is the diagonal matrix of vertex degrees and A is the adjacency matrix. The degree $D_{ii} = \sum_j A_{ij}$. Therefore it can be easily deduced that all rows and columns of the Laplacian matrix L add up to zero. Thus the vector 1 = (1, 1, 1...) is always an eigenvector with eigenvalue zero.

If the network can be separated perfectly into communities, i.e., it can be divided into g non-overlapping groups of vertices G_k(k = 1...g) such that there are edges only within the community and no between-community ones, then the Laplacian will be block diagonal. Each diagonal block will form the Laplacian of its own component, and therefore will have an eigenvector v_k with eigenvalue zero and elements v_k(i) = 1 if i G_k and 0 otherwise. Thus there will be g number of different eigenvectors with eigenvalue 0.[12]

If the network cannot be separated perfectly into communities then the above condition will no longer be perfectly true. Generally there will be the one eigenvector with 1 eigenvalue zero, and g 1 eigenvalues slightly greater than zero, since all eigenvalues of the graph Laplacian are non-negative 1. The corresponding eigenvectors will be given by linear combinations of the eigenvectors v_k as defined above. Therefore, one should be able to find the blocks themselves, at least approximately by looking for eigenvalues of the graph Laplacian only slightly greater than zero and taking linear combinations of the corresponding eigenvectors.

The drawbacks of the spectral bisection method is that it only bisects graphs, i.e, it divides the graph into two partitions. A larger number of community division can be achieved by repeated bisection, but this does not always give satisfactory results. In real world networks, we do not have any prior idea about how many communities are present and how many times the bisection should be performed.

2.2 Hierarchical clustering

In Hierarchical Clustering method[12] a similarity measure that is used to quantify some type of similarity between node pairs is defined. Usually topological similarity is quantified. Various commonly used measures are the cosine similarity, the Jaccard index, and the Hamming distance between rows of the adjacency matrix. Then the similar nodes are grouped into communities according to this measure. There are several common schemes used to group the similar nodes. Single linkage clustering method classifies two groups to be separate if node pairs between the groups have a similarity less than a given threshold value. In complete linkage clustering, all nodes are considered to belong to the same group if they have similarity greater than threshold.

The advantage of hierarchical clustering method is that it does not require the size or number of groups that we have to provide beforehand, therefore, it has been applied to various social networks with predefined similarity metrics, such as the modularity and betweenness measure. However, they are usually slow and the performance highly depends on the corresponding metrics.

2.3 THE MODULARITY MEASURE

It is a property of the network that measures when the division is good, in the sense that there are many edges within the community and only a few between them. The idea is to compare the division to a randomized network with exactly the same vertices and degree in which edges are placed randomly.[4]

Consider a particular division of a network with k communities. The division of the graph into communities can be represented by a k x k symmetric matrix. Each element e_{ij} represents the fraction of edges between the communities i and j. Thus $\sum_i e_{ii}$ gives the fraction of edges that lie within the same community. $\sum_j e_{ij}$ gives the fraction of edges that has atleast one end in community i

- Modularity Q= (number of edges within groups) (expected number of edges within groups).
- $Q = \sum(e_{ii}^2 - a_i^2)$
- e_{ii} = Fraction of edges present within community i.
- a_i = Fraction of edges that have atleast one vertex within community i

2.3.1 Disadvantages

- Requires information about the entire structure of the graph.
- Fails to identify communities smaller than a certain scale.
- Measures the existing links between the nodes but does not consider the absent links between the nodes in the same community.

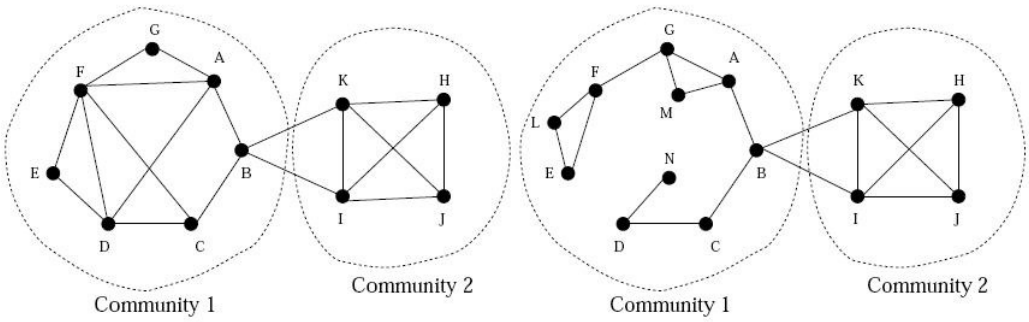

FIGURE 2.1: Two networks with same modularity score but network in the right has more absent links than left one

2.4 Max-Min Modularity

The idea of MM Modularity[3] is based on the innate knowledge that a good division of a network into communities is the one in which not only the number of edges between groups is smaller than expected, but also the one in which the number of unrelated pairs within groups is smaller than expected.

Given a graph G=(V,E) where V is the set of vertices and E is the set of edges, then G'=(V,E') is said to be the complement graph of G if $\forall i, j \in V(i,j) \in E' if and only if (i,j) \notin E$.[3]

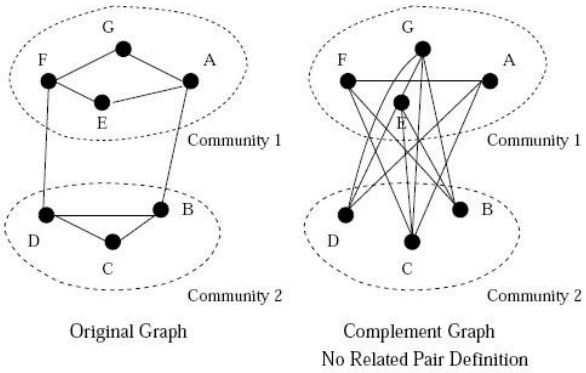

FIGURE 2.2: A graph division and its complement

- Max-min Modularity($Q_{Max-Min}$)= Modularity of Original graph - Modularity of complement graph
- measures the existing links in a community as well as considers the absent links present in the same community.
- cannot detect overlapping communities.

2.5 Clique percolation method

The sequential clique percolation algorithm is an efficient method of detecting overlapping communities in a network.[8, 9, 16] Given a graph G(V,E) where V and E represent the vertices and edges set respectively. S $\subseteq$ G, $\forall$ u,v $\in$ S such that u $\neq$ v and (u,v) $\in$ E, then S si said to be a clique.[8, 16]
S is said to be maximal if there exists no S' such that S $\subset$ S'.

- A k-clique of a graph is a subset of vertices such that the subset is fully connected and there exists an edge between each and every pair of vertex in the subset and size of the subset is k.[8, 16]
- a clique is a fully connected component of a graph.

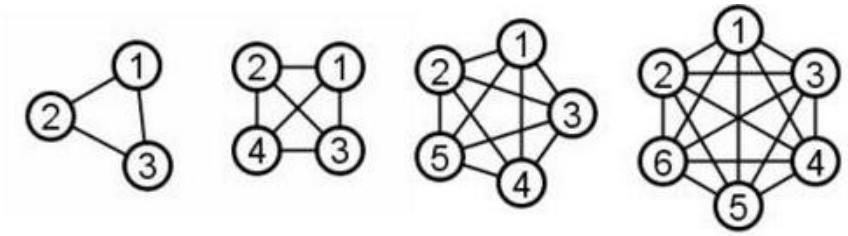

FIGURE 2.3: Examples of cliques

- Two k-cliques are said to be adjacent if the share k-1 nodes in common.
- A K-clique community is a set of all h-cliques($h \geq k$) that are reachable to each other through a series af adjacent k-cliques.

2.5.1 Steps of clique percolation algorithm

- Maximal clique detection. a k-clique is maximal if it is not contained in any other h-clique of h $\geq$ k.
- Create clique-clique overlap matrix. each entry in the matrix indicates the number of common nodes between the respective cliques.
- Replace every element in the matrix greater than k-1 by 1.
- Extract the connected components from the matrix.

2.5.2 Example of clique percolation method

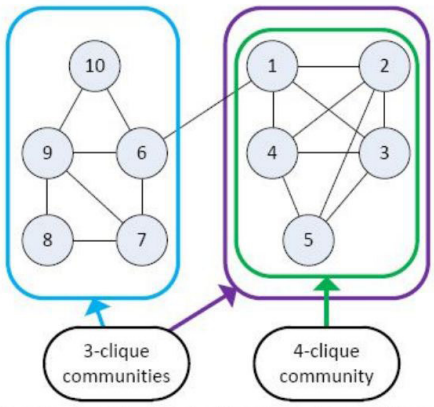

FIGURE 2.4: k-clique communities for k=3 and k=4

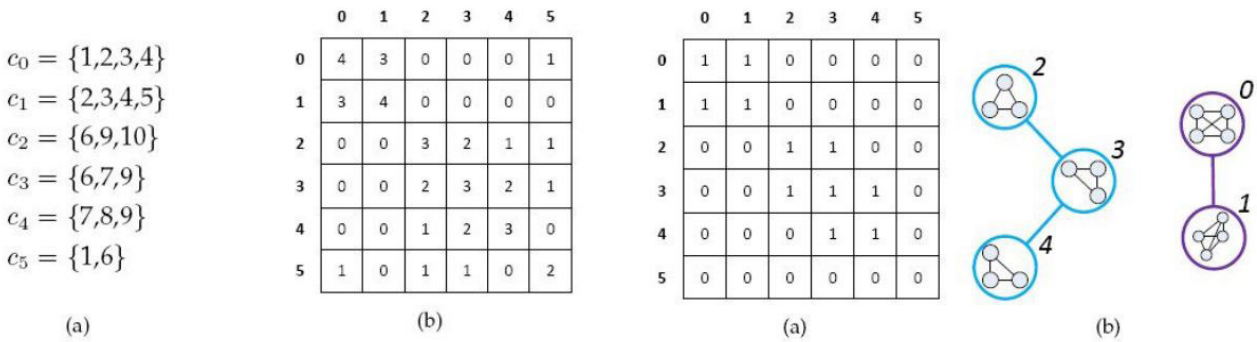

FIGURE 2.5: (a) maximal cliques detected (b)Overlap matrix created (c)Binary matrix created (d)k-clique communities for k=3

2.6 Issues in the existing methods

- The existing graph partitioning methods usually require input parameters such as number of partitions and there size. But it is typically not possible to know the required number of partitions and the partitions in real world cases may not be of same size.

- Modularity based methods require the information regarding the entire structure of the network which is not possible to determine in case of real world large networks such as WWW.

- Modularity based methods are also not able to determine the overlapping communities. but in real world networks, entities or nodes may participate in multiple communities.
- Clique percolation methods require extra space and computation overhead for computing and storing the overlap matrix.

2.7 Objective

The clique percolation method does not require any initial input such as number of partitions or the entire structure of the network. It is also able to detect the overlapping communities in the real world networks.A naive approuch of implementing the clique percolation algorithm would be to generate all the maximal cliques and store them and then compare each of them to find out the connectivity between them. But this would require high computational and space overhead.Our objective in this project is to use a simple backtracking algorithm given by Bron-Kerbosch[2] with minor modifications for generating maximal cliques and avoid generation of sub-maximal and duplicate cliques such that it fits our purpose of implementing clique percolation algorithm and to introduce minor modifications with an aim of reducing the complications in storing the intermediate percolating structures, there by improving the space and computation overhead.

Chapter 3

Algorithimic implementation

3.1 Problem formulation

Let G=(V,E) be a graph where V is the set of vertices and E $\subseteq$ V x V is the set of edges. A k-clique is a subset c $\subseteq$ V such that there exists (i,j) $\in$ E $\forall$ i,j $\in$ c. A k-clique community is the union of all the h-cliques, k $\leq$ h that can be reached by eachother through a series of adjacent k-cliques.

3.2 Data structures used

- input: A[N][N] //Adjacency matrix
- N: number of nodes in the network
- general backtracking algorithm is used to detect maximal cliques.
- Stack S is used to keep track of detected cliques.
- A clique matrix C[][] is used instead of overlap matrix.
- C[i][j]=1 if vertex j belongs to clique i.
- A binary matrix B[][] represents the connected cliques. B[i][j]=1 if cliques i and j are connected.
- M = A; //copy adjacency matrix to a temporary matrix

3.3 Description of algorithm

A maximal clique is a clique that is not contained within any other clique. In order to detect the maximal cliques in the input graph we will use Bron-Kerbosch[2] backtracking algorithm. It is a recursive algorithm and is dependant on three sets.

- Set of nodes that have already been defined as a part of the clique.
- Set of the nodes that are connected to all the nodes of the previous set.
- Set of nodes that have already lead to a valid clique formation and not to be touched again.

I have achieved these three sets by using a stack S that stores the nodes of the clique currently being constructed, a set *neighbour* that holds neighbours of the current node being processed and a set *processed* that holds the nodes that have already been processed.
The set of the above listed sets is represented by the stack S in our implementation. The second set is computed recursively by

- $N = neighbour_i \bigcap neighbour_j \bigcap ... \bigcap neighbour_k$

where i,j...are part of the current clique and k is the current node being processed. The third set is represented by the set

- $not = N \bigcap processed.$

The detailed process if initialization, Clique generation and detection of the connected components are given in the subsequent subsections.

3.3.1 Process initialization

The process is initialized by Algorithm 1 and Algorithm 2 is recursively called to generate the cliques. Algorithm 1 initializes the process by assigning the 1st node to the stack. It computes the neighcour set of the node and the *not* set by the formula $not = neighbour \bigcap processed$ and passes them as argument to Algorithm 2. After the completion of the recursion tree for a particular node, all the maximal cliques containing that particular node are obtained. Algorithm 1 puts the node in the *processed* set and begins the recursion process for the next node by calling Algorithm 2. After the completion of the iteration for all the nodes in the network all the maximal cliques would have been generated and stored in the Clique matrix C.

Algorithm 1: Clique percolation

input : A[N][N] where A is the adjacency matrix of the network

output: Set of maximal cliques stored in clique matrix C and the connection between them Stored in the binary matrix B

```
top ← -1;
Processed ← null;
neighbour ← null;
not ← null;
k ← 0 ;
for i ← 1 to N do
    neighbour ← {neighbours of i}-Processed ;          // neighbour set
    initialization
    not ← Processed ⋂ {neighbours of i};          // not set initialization
    top ← top +1;
    S [top ]← i ;                                 // node entered into stack
    cliqueDetect(neighbour,not);
end
```

3.3.2 Clique Detection

The algorithm recursively calculates the Neighbour set N and the *not* set for each of the candidate nodes forwarded by Algorithm 1 and calls itself till the stopping criteria is satisfied. The algorithm stops when set N and *not* are null. This condition shows that a maximal clique has been detected. The contents of the stack are stored in the Clique matrix C and the algorithm returns one step back. If the set$N$ is null but *not* is not null, the clique so formed is not maximal and is discarded.

Algorithm 2: cliqueDetect(neighbour,not)

```
Processed ← null;
if neighbour = null and not = null then // termination condition
satisfied
    t ← top;
    while t ≥ 0 do // stack content stored in clique matrix
        C [k ][S [t ]] ← 1;
        t ← t-1;
    end
    k ← k +1;
    // row number of clique matrix incremented
end
for ∀j ∈ neighbour do
    top ← top +1;
    S [top ] ← j;
    cliqueDetect(neighbour ⋂ {neighbours of j} - Processed,not ⋂ {neighbours
    of j}); // candidate nodes and not set passed to next level
    Processed ← Processed ⋃ j;
    not ← not ⋃ j;
end
top ← top-1;
return;
```

3.3.3 Determining connected components

After all the maximal Cliques have been detected and stored Algorithm 2 computes the degree of overlapping among the cliques. Degree of overlapping between two cliques i,j is calculated by adding the product of the corresponding row elements . The algorithm creates the binary matrix B, where B[i][j]=1 if cliques i and j have more than k nodes in common.

Algorithm 3: connectClique()

```
sum ← 0;
// degree of overlapping of cliques calculated
for i ← 1 to N do
    for j ← i + 1 to N do
        for k ← 1 to N do
            sum ← sum + C [i][k] * C [j][k];
        end
        if sum > k then // if no.of common nodes is greater than k then
        i,j are connected
            B [i][j] ← 1;
        end
    end
end
```

Chapter 4

Simulations and Results

4.1 A step by step example

Let us go through a step by step example of the whole process so that we can get a clear cut idea about how the process of detecting the cliques and distinguishing the connected components among them actually works.

4.1.1 Adjacency matrix input

```
A =[1  1  0  0  0  0  0  0  0  1;
    1  1  1  1  1  1  1  0  0  1;
    0  1  1  1  0  0  1  0  0  0;
    0  1  1  1  1  1  1  0  0  0;
    0  1  0  1  1  1  1  1  0  0;
    0  1  0  1  1  1  1  1  0  0;
    0  1  1  1  1  1  1  1  1  1;
    0  0  0  0  1  1  1  1  1  1;
    0  0  0  0  0  0  1  1  1  1;
    1  1  0  0  0  0  1  1  1  1;]
```

(a)

```
c1={2,4,5,6,7}
c2={2,3,4,7}
c3={5,6,7,8}
c4={7,8,9,10}
c5={1,2,10}
c6={2,7,10}
```

(b)

FIGURE 4.1: Adjacency matrix of a network with 10 nodes and the cliques detected

4.1.2 Formation of clique matrix

	1	2	3	4	5	6	7	8	9	10
1	0	1	0	1	1	1	1	0	0	0
2	0	1	1	1	0	0	1	0	0	0
3	0	0	0	0	1	1	1	1	0	0
4	0	0	0	0	0	0	1	1	1	1
5	1	1	0	0	0	0	0	0	0	1
6	0	1	0	0	0	0	1	0	0	1

(a)

	1	2	3	4	5	6
1	1	1	1	0	0	1
2	1	1	0	0	0	1
3	1	0	1	1	0	0
4	0	0	1	1	0	1
5	0	0	0	0	1	1
6	1	1	0	1	1	1

(b)

FIGURE 4.2: Corresponding clique matrix(a) and binary matrix(b)

The binary matrix formed shows the cliques that are interconnected with each other for k=3.

4.2 Simulation

In order to examine the complexity of the algorithm, the algorithm was applied to four different undirected and unweighted networks with different number of nodes and edges. The minimum clique size was varied from 3 to 8. The clique number and time required to process was plotted against clique size and the result was observed and analysed.

4.2.1 Simulation 1

- Network used: Dolphin social network[11].
- Number of nodes: 62

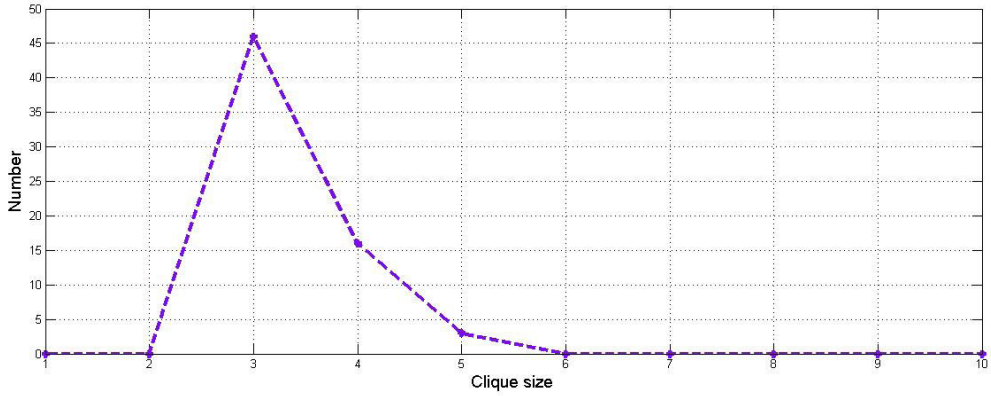

FIGURE 4.3: Clique size vs Number of Dolphin social network

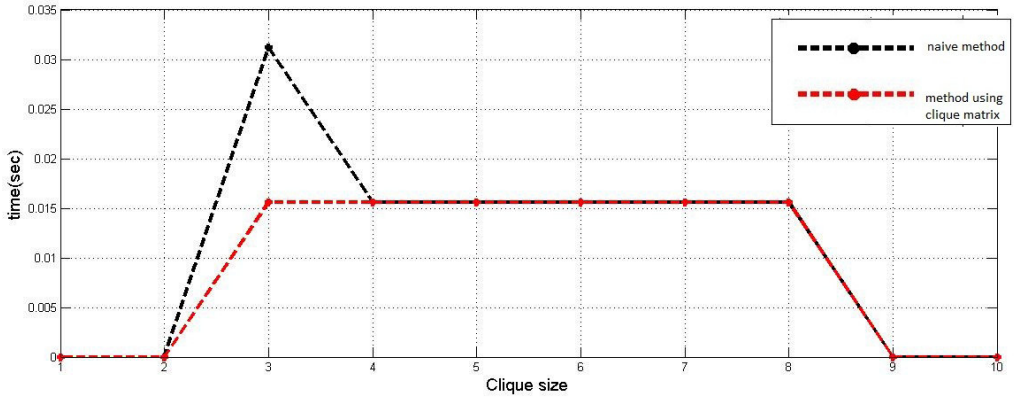

FIGURE 4.4: Clique size vs time Dolphin social network

The algorithm was first applied to the Dolphin social network[11] with 62 nodes. It was observed that maximum number of cliques that is around 47 cliques

were obtained when k=3. The number of cliques gradually reduced as the k value was increased and the clique number became 0 when k value approached 6. So the graph is sparesly dense. The time comparison shows that the algorithm performs better when the number of cliques is more.

4.2.2 Simulation 2

- Network used: Books about US politics[1].
- A network of books about recent US politics sold by the online bookseller Amazon.com. Edges represent frequent co-purchasing of books by same buyer. The network was compiled by V. Krebs.
- Number of nodes: 105

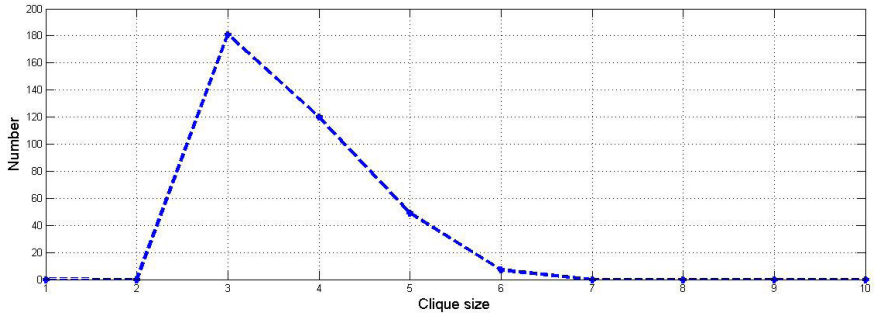

FIGURE 4.5: Clique size vs Number of Books about US politics

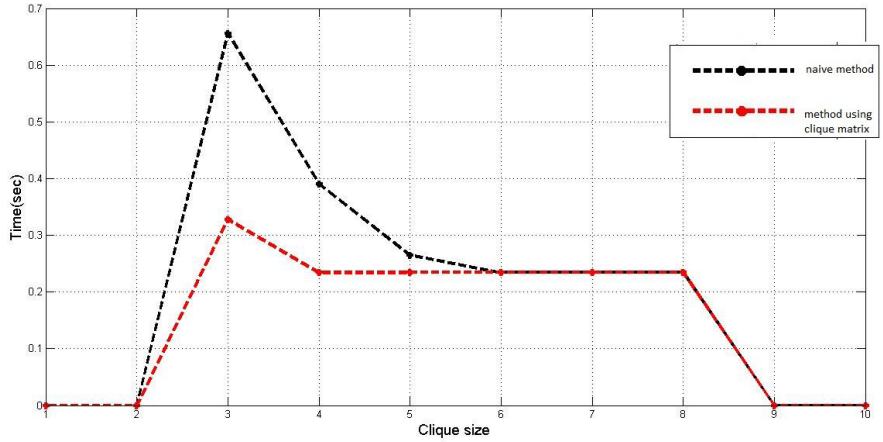

FIGURE 4.6: Clique size vs time of Books about US politics

The algorithm was applied to a second network- Books about US politics[1] with 105 nodes. It was observed that maximum number of cliques that is around

100 cliques were obtained when k=3. The number of cliques gradually reduced as the k value was increased and the clique number became 0 when k value approached 7. So the graph is moderately dense. The time comparison shows that the algorithm performs better when the number of cliques is more. Both the algorithms take same time for clique detection as same method is used, but the new method performs better in determination of connected method.

4.2.3 Simulation 3

- Network used: American College Football[6].
- Network of American football games between Division IA colleges during regular season Fall 2000
- Number of nodes: 115

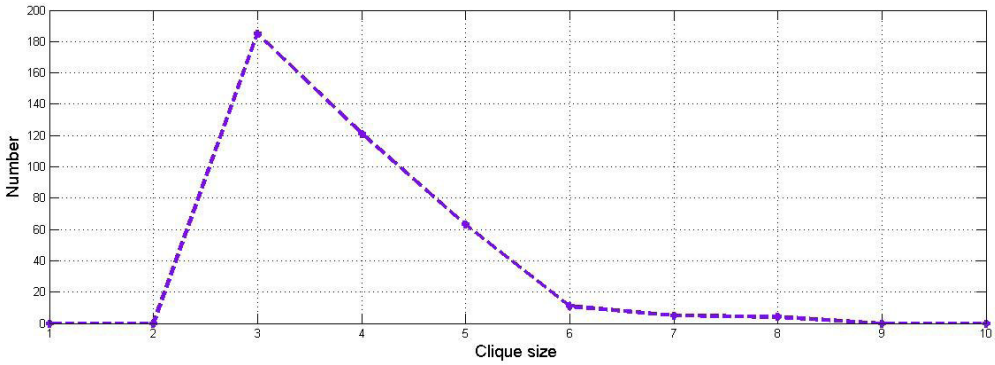

FIGURE 4.7: Clique size vs Number of American College Football network

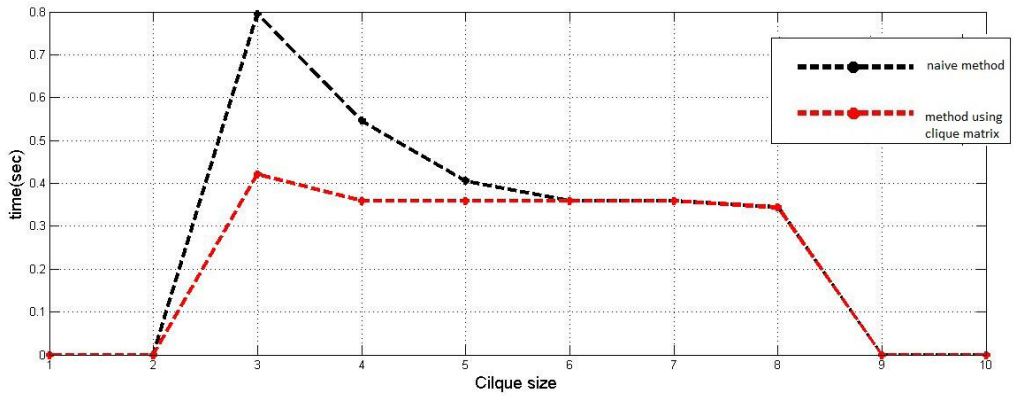

FIGURE 4.8: Clique size vs time of American College Football network

The algorithm was applied to a third network- American College Football[6] with 115 nodes. It was observed that maximum number of cliques that is around 110 cliques were obtained when k=3. The number of cliques gradually reduced as the k value was increased and the clique number approached to 0 when k value approached 8 bt did not reach zero. So the graph is moderately denser than the previous network. The time comparison shows that the algorithm performs better when the number of cliques is more. Both the algorithms take same time for clique detection as same method is used, but the new method performs better in determination of connected method.

4.2.4 Simulation 4

- Network used: Coauthorships in network science[13].
- Coauthorship network of scientists working on network theory and experiment, as compiled by M. Newman in May 200
- Number of nodes: 1589

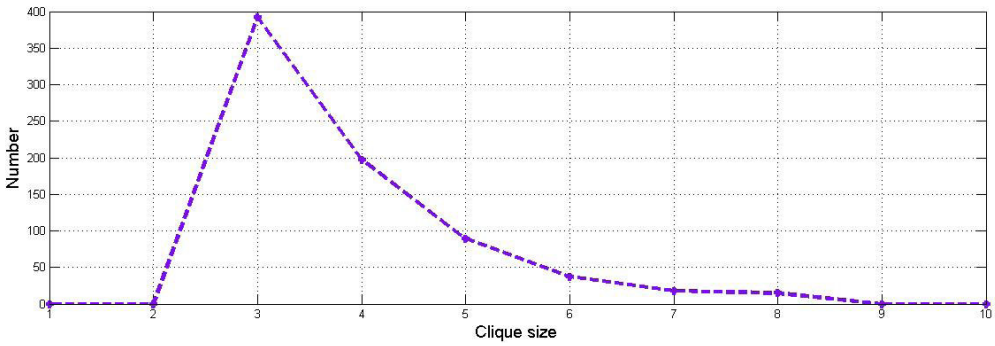

FIGURE 4.9: Clique size vs Number of Coauthorships in network science

The algorithm was applied to a fourth network- Coauthorships in network science[13] with 1589 nodes. It was observed that maximum number of cliques that is around 400 cliques were obtained when k=3. The number of cliques gradually reduced as the k value was increased and the clique number approached to 0 when k value approached 8 bt did not reach zero.The Clique to node ratio is smaller in this case than the previous network. So the graph is sparesly connected than the previous network. The time comparison shows that the algorithm performs better when the number of cliques is more. Both the algorithms take same time

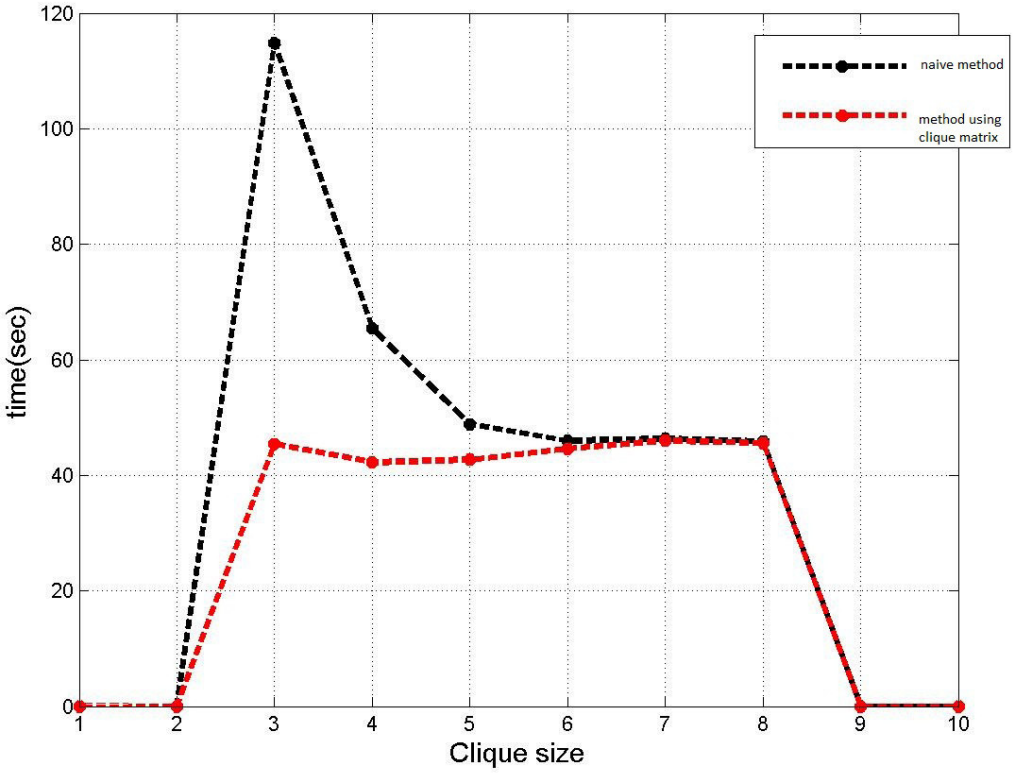

FIGURE 4.10: Clique size vs time of Coauthorships in network science

for clique detection as same method is used, but the new method performs better in determination of connected method.

4.3 Analysis

The above simulations performed show that in all the four cases the number of cliques detected is maximum when clique size is three. The number of cliques reduces as the clique size is increased. So the overlapping nature of the partition formed reduces with the increase in clique size. The comparision of the simulations shows that the algorithm performs better when the number of cliques present is more and has a better time complexity. As the algorithm does not require any extra data structure to store the cliques formed, we can conclude that it requires less space overhead during the computation.

4.4 Conclusions

Community detection is a problem of detecting subgraphs with higher within edge density than between edge density in a network. Communities can be overlapping as in real world social network as the individuals in the network may be involved in several communities. k-clique percolation method is one of the efficient methods for detecting the overlapping communities in a network. We implemented and analysed the k-clique percolation algorithm on different networks with different network structures and varying clique size. K-clique percolation is a hard problem to implement well, due to the difficulty of producing intermidiate representations of percolating structures. The method is challenged by the presence of large number of cliques. We implemented the method by using a Clique matrix that can be used to store the cliques generated and at the same time can represent the degree of overlapping between the cliques there by making the representation and computation of intermidiate structures simpler.

4.5 Future works

The k-clique percolation algorithm depends heavily on the clique detection algorithm being used. The k-Clique percolation algorithm can be improved by using more efficient clique detection algorithms which is which is a current research area. Computation of the degree of overlappiing among the cliques has $O(n^2)$ complexity. It can be reduced by using better data structures to store the intermediate structures.

Chapter 5

Bibliography

[1] Books about us politics. http://networkdata.ics.uci.edu/data.php?d=polbooks.

[2] Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[3] Jiyang Chen, Osmar R Zaïane, and Randy Goebel. Detecting communities in social networks using max-min modularity. *SDM 2009*, pages 978–989, 2009.

[4] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[5] Tanja Falkowski, Anja Barth, and Myra Spiliopoulou. Dengraph: A density-based community detection algorithm. In *Web Intelligence, IEEE/WIC/ACM International Conference on*, pages 112–115. IEEE, 2007.

[6] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[7] Enrico Gregori, Luciano Lenzini, and Simone Mainardi. Parallel k-clique community detection on large-scale networks. 2012.

[8] Jussi M Kumpula, Mikko Kivelä, Kimmo Kaski, and Jari Saramäki. Sequential algorithm for fast clique percolation. *Physical Review E*, 78(2):026109, 2008.

[9] Conrad Lee, Fergal Reid, Aaron McDaid, and Neil Hurley. Detecting highly overlapping community structure by greedy clique expansion. *arXiv preprint arXiv:1002.1827*, 2010.

[10] Zhenping Li, Shihua Zhang, Rui-Sheng Wang, Xiang-Sun Zhang, and Luonan Chen. Quantitative function for community detection. *Physical Review E*, 77(3):036109, 2008.

[11] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.

[12] Mark EJ Newman. Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):321–330, 2004.

[13] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[14] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[15] Clara Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. In *Parallel Problem Solving from Nature–PPSN X*, pages 1081–1090. Springer, 2008.

[16] Fergal Reid, Aaron McDaid, and Neil Hurley. Percolation computation in complex networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 274–281. IEEE, 2012.

[17] Erin N Sawardecker, Marta Sales-Pardo, and Luís A Nunes Amaral. Detection of node group membership in networks with group overlap. *The European Physical Journal B*, 67(3):277–284, 2009.

[18] Karsten Steinhaeuser and Nitesh V Chawla. Community detection in a large real-world social network. In *Social Computing, Behavioral Modeling, and Prediction*, pages 168–175. Springer, 2008.